

APPROXIMATION RATIO OF LD ALGORITHM FOR MULTI-PROCESSOR SCHEDULING AND THE COFFMAN–SETHI CONJECTURE

PERUVEMBA SUNDARAM RAVI, LEVENT TUNÇEL

ABSTRACT. Coffman and Sethi proposed a heuristic algorithm, called LD, for multi-processor scheduling, to minimize makespan over flowtime-optimal schedules. LD algorithm is a natural extension of a very well-known list scheduling algorithm, Longest Processing Time (LPT) list scheduling, to our bicriteria scheduling problem. Moreover, in 1976, Coffman and Sethi conjectured that LD algorithm has precisely the following worst-case performance bound: $\frac{5}{4} - \frac{3}{4(4m-1)}$, where m is the number of machines. In this paper, utilizing some recent work by the authors and Huang, from 2013, which exposed some very strong combinatorial properties of various presumed minimal counterexamples to the conjecture, we provide a proof of this conjecture. The problem and the LD algorithm have connections to other fundamental problems (such as the assembly line-balancing problem) and to other algorithms.

1. INTRODUCTION

The most fundamental machine environment in multiprocessor scheduling problems is a *parallel identical machine model*. In this basic set-up, we have m parallel identical machines that are simultaneously available at time zero, and n independent jobs, all simultaneously available at time zero, indexed by $1, 2, \dots, n$ with given processing times p_1, p_2, \dots, p_n . No pre-emption is allowed, and the machines are assumed to be completely reliable. For a scheduling problem environment described above, with data m, p_1, p_2, \dots, p_n , there are two performance criteria that immediately come to mind:

- minimize the completion time of the last job (i.e., *makespan*),
- minimize the total (or equivalently the average) time that the jobs spend in the system (i.e., total or average *flowtime*).

Given a feasible schedule, let C_j denote the completion time of job j in that schedule. By denoting $C_{\max} := \max_{j \in \{1, 2, \dots, n\}} \{C_j\}$, our two criteria are:

- minimize C_{\max} ,
- minimize $\sum_{j=1}^n C_j$.

Both of these objective functions are easily justifiable. Indeed, the minimization of makespan may ensure optimal utilization of resources (i.e., machines) as well as ensuring the earliest

Date: May 6, 2015.

Key words and phrases. parallel identical machines, makespan, total completion time, total flowtime, approximation algorithms, multi-processor scheduling, bicriteria scheduling problems.

Peruvemba Sundaram Ravi: Operations and Decision Sciences, School of Business and Economics, Wilfrid Laurier University, Waterloo, Ontario N2L 3C5, Canada (e-mail: pravi@wlu.ca)

Levent Tunçel: Department of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Waterloo, Ontario N2L 3G1, Canada (e-mail: ltuncel@uwaterloo.ca).

possible start times for other tasks that require the completion of all the jobs $1, 2, \dots, n$ to be started. Minimization of total flow time $F := \sum_{j=1}^n C_j$, minimizes the amount of time the jobs spend in the system (in our setting this is C_j for each job j). Thus, minimizing F equivalently minimizes, in many applications, work-in-process inventory. A feasible schedule is called *flowtime-optimal* if it minimizes F . In this paper, we consider the bicriteria optimization problem of minimizing makespan among all flowtime-optimal schedules. In scheduling theory notation, let F^* denote the optimal objective function value of $Pm / / \sum C_j$. Then, our bicriteria optimization problem is: $Pm / / C_{\max}; \sum C_j = F^*$, which we call *Flowtime-Makespan (FM)* problem.

There are two single objective function scheduling problems that make up our bicriteria optimization problem: $Pm / / C_{\max}$, and $Pm / / \sum C_j$. The second problem is as easy as sorting and, as a result, admits algorithms with $O(n \log(n))$ complexity. Moreover, we have a complete characterization of all optimal solutions of $Pm / / \sum C_j$. Conway, Maxwell and Miller [7], in their seminal book, develop the notion of *rank* for the FM problem. To simplify presentation, we may assume that m divides n (otherwise, we can add $(m \lceil n/m \rceil - n)$ dummy jobs with zero processing times; this modification does not affect the conclusions of this paper). Then, in such an instance, there are $k := n/m$ ranks. We may further assume that the jobs are indexed in nonincreasing order of processing times (i.e., with job 1 having the largest processing time), the set of jobs belonging to rank r are the following: $(r-1)m+1, (r-1)m+2, \dots, (r-1)m+m$. Then a feasible schedule is *flowtime-optimal* if jobs are assigned in decreasing order of ranks, with the jobs in rank 1 being assigned last. Since within each rank the assignment of jobs to machines can be arbitrary, it immediately follows that there are at least $(m!)^{\lfloor n/m \rfloor}$ flowtime-optimal schedules. From a mathematical viewpoint, it makes sense to consider a secondary criterion to choose a “best” flowtime-optimal schedule among these huge number of schedules everyone of which minimizes total flowtime. Also, this is reasonable from a practical viewpoint.

The first problem, $Pm / / C_{\max}$, is \mathcal{NP} -hard even for $m = 2$ (trivial reduction from PARTITION). Graham’s ground-breaking work on the subject in 1960’s tackled the problem $Pm / / C_{\max}$. This work was ground-breaking not only in approximation algorithms for scheduling, but in approximation algorithms in general. Graham first proved:

Theorem 1. (Graham [13]) *List Scheduling algorithm has worst-case approximation ratio of $(2 - \frac{1}{m})$. Moreover, this bound is achievable for every $m \geq 2$.*

Then, Graham analyzed the List Scheduling algorithm when the list is given in LPT order and provided a very elegant proof of the following result:

Theorem 2. (Graham [14]) *LPT-List Scheduling algorithm has worst-case approximation ratio of $(\frac{4}{3} - \frac{1}{3m})$. Moreover, this bound is achievable for every $m \geq 2$.*

Just like the scheduling problem $Pm / / C_{\max}$, the FM problem is also \mathcal{NP} -hard (a result of Bruno, Coffman and Sethi [1]). In 1976, Coffman and Sethi [5] proposed some approximation algorithms for the FM problem. Among these algorithms, in the LD algorithm (which is an extension of the LPT list scheduling for the FM problem), ranks are assigned in increasing order, starting with rank one, which is the rank containing the set of m jobs with the largest

processing times. Jobs within the same rank are assigned largest-first onto distinct machines as the machines become available after completing the jobs in the previous ranks. After all the jobs are assigned, the schedule is reversed and all jobs in the last rank (that is, rank k) must be set to the same starting time of zero. Coffman and Sethi conjectured the following worst-case bound for the LD algorithm.

Coffman-Sethi conjecture [5]:

the LD algorithm has a makespan ratio with a worst-case bound equal to

$$\frac{5m-2}{4m-1} = \frac{5}{4} - \frac{3}{4(4m-1)}.$$

The authors and Huang [21] constructed the following family of instances, proving that the above conjectured ratio cannot be improved for any $m \geq 2$. For every integer $m \geq 2$, let $n := 3m$ and define the processing times as:

$$p_j := \begin{cases} 0, & \text{for } j \in \{1, 2, \dots, m-1\}; \\ m, & \text{for } j = m; \\ (j-1), & \text{for } j \in \{m+1, m+2, \dots, 2m\}; \\ (j-2), & \text{for } j \in \{2m+1, 2m+2, \dots, 3m\}. \end{cases}$$

It is easy to verify that the ratio of the objective value of an LD schedule to the optimal objective value is exactly $\frac{5m-2}{4m-1}$, for every integer $m \geq 2$. In Figure 1, we present an LD schedule for the three-machine instance of this family of bad instances. Note that each of the machines 1 and 2 have a job (either job 1 or job 2) with processing time zero, started and finished at time zero (not shown on the Gantt chart). Completion times are 10, 10, and 13 on the machines 1, 2, and 3 respectively. Figure 2 presents the Gantt chart for an optimal schedule, with optimal

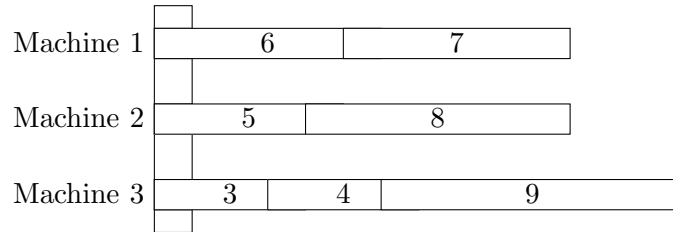
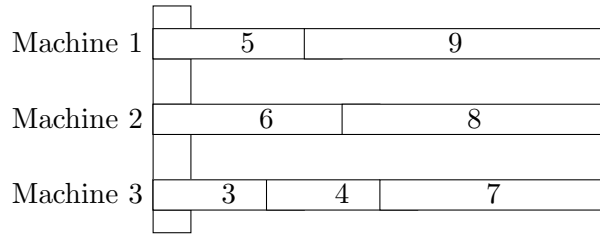


FIGURE 1. LD schedule for $m = 3$

makespan 11, for the same three-machine instance. Again, machines 1 and 2 have jobs with zero processing times. Further note that this schedule is “rectangular,” providing an obvious certificate of the optimality of the underlying makespan. This term “rectangular schedule,” is rigorously defined in the next section.

The Coffman-Sethi conjecture has remained open for nearly four decades. The first major progress was made by the authors and Huang in [21]. In the next section, utilizing our recent work with Huang [21], we provide a proof of this conjecture. Our approach in obtaining a proof of Coffman and Sethi’s conjectured bound is to identify properties of a hypothesized

FIGURE 2. Optimal schedule for $m = 3$

minimal counterexample to the conjecture. To define our notion of minimality, we use a five-level hierarchical ordering of various attributes of the FM problem instances. Using integrality of the data and some more sophisticated properties of the minimal counterexamples, we construct some “smaller” problem instances by subtracting nonnegative integers from the number of machines, the number of jobs, or the integer-valued processing times. This exposes some very strong combinatorial structures of presumed minimal counterexamples. These special combinatorial structures lead us to very strong constraints that capture the fact that all problem instances that are “smaller” than the minimal counterexample at hand, must satisfy the conjecture. This eventually leads to a contradiction for problem instances with four or more ranks. Given that a minimal counterexample cannot exist for one, two or three ranks, or with two or three machines as was previously proved by the authors and Huang [21], these results prove that the conjecture holds for all instances of problem FM.

Note that there are more sophisticated algorithms with better approximation guarantees for the kind of scheduling problems (and in general combinatorial optimization problems) that we are considering in this paper. The bin-packing problem provides another combinatorial optimization context which is helpful in attacking makespan minimization problems in multi-processor scheduling. First Fit Decreasing (FFD) algorithm [18] (also see [19]) admits good approximation bounds. Various alternatives to FFD were proposed [12, 4]. Moreover, Coffman, Garey and Johnson [3] designed the MultiFit algorithm (an approximation algorithm for minimizing makespan) based on FFD and proved a worst-case performance bound. Recently the exact performance bound of MultiFit was established as $24/19$ by Hwang and Lim [17]. de la Vega and Lueker [22] propose a linear-time approximation scheme for the bin packing problem. For some other work on MultiFit algorithm and related topics, see [2, 8, 9, 11, 23]. Hochbaum and Shmoys [15] propose a PTAS (polynomial-time approximation scheme) for the makespan minimization problem on the parallel identical machine model. Eck and Pinedo [10] propose a new algorithm LPT* for the FM problem and for the two machine case, prove the worst-case approximation ratio of $28/27$. A slight generalization of the FM problem can be formulated as the problem of permuting the elements within the columns of a m -by- n matrix with nonnegative entries, so as to minimize its maximum row sum. This problem, which models the assembly line balancing problem, was studied by Coffman and Yannakakis [6] as well as Hsu [16]. Therefore, FM problem is a fundamental problem in multi-processor scheduling, with close ties to other problems in combinatorial optimization. The LD algorithm is also closely related to some fundamental heuristics for such problems.

In the next section, we provide a detailed, rigorous definition of the LD algorithm. In Section 3, we present a proof of the Coffman–Sethi conjecture.

2. LD ALGORITHM, DEFINITIONS AND SOME PROPERTIES

Let us index the jobs in nonincreasing order of processing times. The set of jobs belonging to rank r is:

$$\{(r-1)m+1, (r-1)m+2, \dots, (r-1)m+m\}.$$

A feasible schedule in which all rank $(r+1)$ jobs are started before all rank r jobs (where $r \in \{1, 2, \dots, (n/m) - 1\}$) is said to satisfy the *rank restriction* or *rank constraint*. A feasible schedule without idle time and satisfying the rank constraint, and in which all rank n/m jobs start at time zero, is a *flowtime-optimal schedule*.

In every rank r , we identify the largest and smallest processing times and denote them by λ_r and μ_r . Therefore, we have

$$(1) \quad \lambda_1 \geq \mu_1 \geq \lambda_2 \geq \mu_2 \geq \dots \geq \lambda_{k-1} \geq \mu_{k-1} \geq \lambda_k \geq \mu_k \geq 0,$$

where, $k := \lceil \frac{n}{m} \rceil$. The *profile* of a schedule after rank r is defined as the sorted set of completion times (in nonincreasing order) on m machines after rank r . When all the jobs in first r ranks (out of a total of k ranks) have been assigned to machines, and the jobs in the remaining ranks have not yet been assigned to machines, the profile after rank r is called the *current profile*. We denote the current profile by $a(r) \in \mathbb{R}^m : a_1(r) \geq a_2(r) \geq \dots \geq a_m(r)$. I.e., $a_i(\ell)$ denotes the i th largest completion time in the schedule, after rank ℓ .

The LD algorithm of Coffman and Sethi [5] schedules the ranks in the following order: $1, 2, \dots, k-1, k$. Denote the current profile by $a \in \mathbb{R}^m : a_1 \geq a_2 \geq \dots \geq a_m$. Then, schedule the jobs in the next rank so that the job with the largest processing time is matched with a_m (the smallest part of the current profile), second largest processing time is matched with a_{m-1} (the second smallest part of the current profile), etc., and the smallest processing time is matched with a_1 (the largest part of the current profile). After all the jobs are scheduled, the schedule is reversed and left-justified (i.e., we start the first job on each machine at time zero).

As we mentioned in the introduction, without loss of generality, we may assume the property

$$(\text{Property.1}) \quad n = mk,$$

while allowing some jobs to have a zero processing time. Given an instance of the FM problem, we denote by t_{LD} the makespan of the LD schedule(s). We use t^* to denote the makespan of the optimal schedule(s). The following lemma is stated without proof. A proof is provided in [20].

Lemma 1. (See [21, 20]) *For the FM problem and the LD algorithm, the following must hold: If there exists a counterexample to a conjectured t_{LD}/t^* ratio, then there exists a counterexample with integer processing times.*

We will, therefore, restrict our attention to problem instances with integer data in this paper. Note that due to integrality of the processing times, the smallest nonzero processing time is bounded below by one. We define the ordered set of processing times P for a scheduling problem instance with m machines and k ranks to consist of elements equal to the processing times of these mk jobs arranged in nonincreasing order. We use $P(j)$ to refer to the j^{th} entry of P .

In our characterizations of minimal counterexamples to the Coffman–Sethi conjecture, minimality is defined based on the attributes: the number of machines, the number of ranks, and the ordered set of processing times with respect to a five-level, hierarchical grading. See [21] for the definition minimality for the classes of counterexamples we consider.

Throughout this paper, we will isolate many useful properties of minimal counterexamples of various types. Note that, as it was established in [21], we may assume that in a minimal counterexample, the following property holds:

$$\text{(Property.2)} \quad \mu_r = \lambda_{r+1}, \forall r \in \{1, 2, \dots, k-1\} \text{ and } \mu_k = 0.$$

We continue with some more definitions.

- A problem instance of *Type I* is a problem instance that has integer processing times.

A counterexample of Type I is an FM problem instance of that type which violates the Coffman–Sethi conjecture. A *minimal counterexample* of Type I is a counterexample of that type for which there does not exist a smaller counterexample (based on the notion of minimality defined in [21]) of the Type I.

A *rectangular schedule* is a feasible schedule in which all machines are busy between time zero and the makespan. Note that every rectangular schedule minimizes the makespan, since its objective function value matches an obvious lower bound of $\sum_{j=1}^n p_j/m$ on the makespan of every feasible schedule.

- A problem instance of *Type IR* is one with integer processing times and a rectangular optimal schedule.

A counterexample of Type IR is defined analogously.

3. A PROOF OF THE COFFMAN–SETHI CONJECTURE

We state without proofs the following two lemmas from previous work.

Lemma 2. (*Ravi, Tunçel and Huang [21]*) *An increase in one or more processing times of jobs in rank r for $r \in \{1, 2, \dots, k-1\}$ (with no change in the remaining processing times, and subject to the rank constraint) does not result in a reduction in any element of the profile $a(\ell)$ of an LD schedule after rank $\ell \in \{r, r+1, \dots, k\}$.*

Lemma 3. (*Ravi, Tunçel and Huang [21]*) *If the Coffman–Sethi conjecture is false, then there exists a minimal counterexample to the conjecture of Type IR.*

Next, we describe two useful ways (procedures REDUCE($P1, r$) and \square -REDUCE($P1, r$)) of generating “smaller” FM instances from a given FM instance. Let $P1$ denote an FM problem instance.

REDUCE($P1, r$): Construct $P2$ from $P1$ by subtracting one time unit from the processing time of every job in rank $r-1$ and subtracting one time unit from the processing time of every job in rank r that has a processing time of λ_r . Leave the remaining processing times unchanged.

Figure 3 presents an LD schedule S for $m := 3$, $P1 := [9, 8, 7, 7, 6, 5, 5, 2, 1]$. Completion times are 15, 16, and 19 on the machines 1, 2, and 3 respectively.

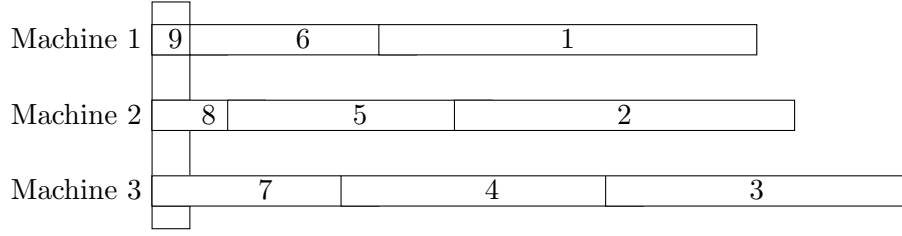
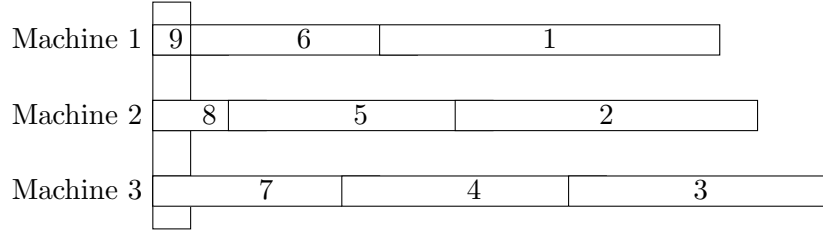
FIGURE 3. An LD schedule S for the instance given by $P1$

Figure 4 presents the result of the application of $\text{REDUCE}(P1,2)$ to the original LD schedule S , yielding $\text{REDUCE}(P1,2) = [8, 7, 6, 6, 6, 5, 5, 2, 1]$, and the completion times become 14, 15, and 17 on the machines 1, 2, and 3 respectively.

FIGURE 4. LD schedule S_1 for the instance $\text{REDUCE}(P1,2)$

\square - $\text{REDUCE}(P1,r)$: Construct $P2$ from $P1$ by applying the procedure $\text{REDUCE}(P1,r)$ to $P1$. Construct $P2R$ from $P2$ as follows. For every job in rank 1 of the optimal schedule for $P2$ that is processed on a machine with a completion time after rank k that is less than the makespan, increase the processing time so that the completion time after rank k becomes equal to the makespan.

Note that every instance generated by \square - $\text{REDUCE}(P1,\cdot)$ has, by construction, a rectangular optimal schedule. Figure 5 presents an optimal schedule for the instance given by $P2 := [8, 7, 6, 6, 6, 5, 5, 2, 1]$. In this optimal schedule, the completion times are 15, 15, and 16 on the machines 1, 2, and 3 respectively.

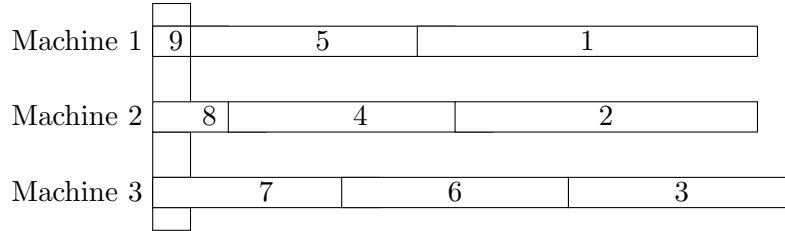
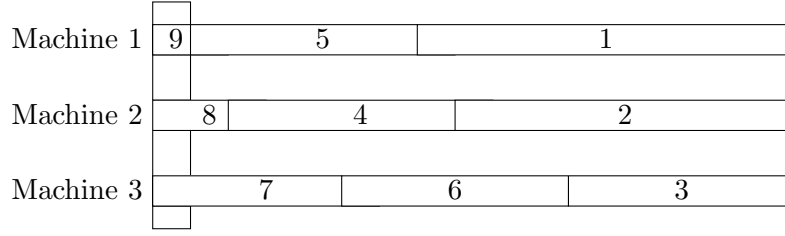
FIGURE 5. An optimal schedule for the instance given by $P2$

Figure 6 presents an optimal schedule for the instance \square - $\text{REDUCE}(P1,2)$. Here, \square - $\text{REDUCE}(P1,2) = [9, 8, 6, 6, 6, 6, 5, 5, 2, 1]$ whose optimal schedules yield a makespan of 16 on every machine.

FIGURE 6. An optimal schedule for the instance \square -REDUCE(P1,2)

Proposition 1. (1) Let $P1$ be a minimal counterexample of Type I and $P2 := \text{REDUCE}(P1, r)$. Then, $t_{LD}(P2) \leq t_{LD}(P1) - 2$.
 (2) Let $P1$ be a minimal counterexample of Type IR and $P2R := \square\text{-REDUCE}(P1, r)$. Then, $t_{LD}(P2R) \leq t_{LD}(P1) - 2$.

Proof. We prove the second assertion only (the proof of the first assertion is similar). For the purpose of a contradiction, assume that the proposition is false. Therefore, the LD makespan of $P2R$ is at least $t_{LD} - 1$, where t_{LD} denotes the makespan of the LD schedule for $P1$. It is evident that $P2R$ has an optimal makespan that is at most $t^* - 1$, where t^* denotes the optimal makespan of $P1$. Therefore $P2R$ is a problem instance of Type IR with a worse approximation ratio than $P1$ and the same number of ranks and machines and jobs with nonzero processing times as $P1$. Thus, we have a contradiction and therefore the proposition must be true. \square

We define a minimal counterexample of Type IR1 as follows. If the Coffman–Sethi conjecture is false, a minimal counterexample of Type IR1 is a minimal counterexample of Type IR that has an LD schedule with the property that every machine with a completion time after rank k equal to the makespan has a job with processing time equal to λ_k in rank k . (Recall, k denotes the number of ranks.)

Lemma 4. (Ravi, Tunçel and Huang [21]) *If the Coffman–Sethi conjecture is false, then there exists a minimal counterexample to the conjecture of Type IR1, and every minimal counterexample of Type IR is a minimal counterexample of Type IR1.*

Below, we give a proof of this lemma utilizing the procedure REDUCE and \square -REDUCE to give a glimpse of how to establish properties of Type IR1 and other related minimal counterexamples.

Proof. Suppose the Coffman–Sethi conjecture is false. Then, by Lemma 3, a counterexample of Type IR exists. Suppose, for the purpose of a contradiction that there exists a minimal counterexample $P1$ of Type IR that is not Type IR1. Now construct a problem instance $P2R$ of Type IR by applying the procedure \square -REDUCE($P1, k$) to $P1$. The first step of the procedure is REDUCE($P1, k$) which produces a problem instance $P2$. This step leaves the assignment of jobs in each rank to machines in the LD schedule unchanged. Therefore $P2$ has an LD makespan of $t_{LD} - 1$, where t_{LD} denotes the LD makespan of $P1$. By Lemma 2, the LD makespan of $P2R$ cannot be less than the LD makespan of $P2$. Therefore it is greater than or equal to $t_{LD} - 1$. However, by Proposition 1, the LD makespan of $P2R$ is less than or equal to $t_{LD} - 2$. Therefore, we have a contradiction and the lemma must hold. \square

Our proof of the main result of this paper uses the concept of Type I2 instances as established in [21]. The proof that “if Coffman–Sethi conjecture is false then a counterexample of Type I2 exists” uses procedures like REDUCE and \square -REDUCE and the properties of Type IR1 counterexamples. Next, we define Type I2 instances.

If the Coffman–Sethi conjecture is false, a counterexample to the conjecture of *Type I2* is a counterexample of Type I that has an LD schedule with the following properties:

- (i) It has only one machine i' with a completion time after rank k equal to the makespan.
- (ii) Machine i' has a processing time equal to λ_r in rank r for every $r \in \{2, 3, \dots, k\}$.

Lemma 5. (*Ravi, Tunçel and Huang [21]*) *If the Coffman–Sethi conjecture is false, then there exists a minimal counterexample to the conjecture of Type I2. Moreover, in a minimal counterexample of Type I2, the following properties hold:*

- the sole machine i' with a completion time after rank k equal to the makespan in the LD schedule has a processing time equal to μ_1 in rank 1;
- there exists at least one machine i'' with $i'' \neq i'$, such that the completion time after rank $(k - 1)$ on machine i'' is greater than or equal to the completion time after rank $(k - 1)$ on machine i' ;
- the smallest completion time after rank k on any machine is at least

$$t_{LD} - \max_{r \in \{2, 3, \dots, k\}} \{\lambda_r - \mu_r\}.$$

Proof. See Lemmas 6, 7, 8, 9, and 10 of [21]. \square

Note that Lemma 5 exposes many, combinatorially very strong properties of a minimal counterexample of Type I2. These properties allow us to have access to very strong inequalities on the optimal makespan in terms of the makespan of an LD schedule for such minimal counterexamples.

Theorem 3. (*Ravi, Tunçel and Huang [21]*) *The Coffman–Sethi conjecture holds for all instances with either property given below:*

- (i) $m \leq 3$ (FM instances with at most three machines),
- (ii) $k \leq 3$ (FM instances with at most three ranks, i.e., for all machine-job pairs (m, n) satisfying $n \leq 3m$).

Proof. See, respectively, Theorems 2 and 3 of [21]. \square

Next, we prove that the conjecture holds for all the remaining cases.

Theorem 4. *The Coffman–Sethi conjecture holds for all instances with either property given below:*

- (i) $m \geq 4$ (FM instances with at least four machines),
- (ii) $k \geq 4$ (FM instances with at least four ranks, i.e., for all machine-job pairs (m, n) satisfying $n \geq 3m + 1$).

Proof. Suppose the above claim is false. Then there exists a minimal counterexample of Type I2 to the Coffman–Sethi conjecture. Since the conjecture holds for all instances with $m \leq 3$ as

well as for all instances with $k \leq 3$ (by Theorem 3), there must exist a minimal counterexample of Type I2 to the claim with k and m both at least equal to four. Let t denote the makespan for an LD schedule of the minimal counterexample of Type I2 with k ranks. Then, by Lemma 5, we have

$$mt^* \geq t + (t - \lambda_k) + (m - 2) \left(t - \max_{r \in \{2, \dots, k\}} \{\lambda_r - \mu_r\} \right).$$

The last relation is equivalent to

$$(2) \quad t^* \geq t - \frac{\lambda_k}{m} - \left(1 - \frac{2}{m} \right) \max_{r \in \{2, \dots, k\}} \{\lambda_r - \mu_r\}.$$

Since we are working with a counterexample,

$$(3) \quad t > \left(\frac{5m - 2}{4m - 1} \right) t^*.$$

Inequalities (2) and (3) imply,

$$(4) \quad \left(\frac{m - 1}{4m - 1} \right) t^* < \frac{\lambda_k}{m} + \left(1 - \frac{2}{m} \right) \max_{r \in \{2, \dots, k\}} \{\lambda_r - \mu_r\}.$$

Suppose that the maximum, $\max_{r \in \{2, \dots, k\}} \{\lambda_r - \mu_r\}$, is attained by $r = k$. Then, (4) implies (since $\mu_k = 0$): $t^* < (4 - \frac{1}{m}) \lambda_k$. However, $t^* \geq \lambda_k + \sum_{r=1}^{k-1} \mu_r > k \lambda_k$. Since $k \geq 4$, we reach a contradiction. Therefore, we may assume, there exists $s \in \{2, 3, \dots, k - 1\}$ such that $\max_{r \in \{2, \dots, k\}} \{\lambda_r - \mu_r\} = \lambda_s - \mu_s$.

Using (2) and (3), as well as the facts $\mu_s = \lambda_{s+1}$ and $\lambda_k \leq \lambda_{s+1}$, we obtain

$$(5) \quad \left(\frac{m - 1}{5m - 2} \right) t < \frac{\lambda_{s+1}}{m} + \left(1 - \frac{2}{m} \right) (\lambda_s - \lambda_{s+1}).$$

Since

$$(6) \quad t = 2\lambda_2 + \sum_{r=3}^k \lambda_r,$$

substituting this for t in (5), we derive:

$$(7) \quad \lambda_2 < -\frac{1}{2} \sum_{r=3}^k \lambda_r + \frac{1}{2(m-1)} \left(5 - \frac{2}{m} \right) \lambda_{s+1} + \left[\frac{m-2}{2(m-1)} \right] \left(5 - \frac{2}{m} \right) (\lambda_s - \lambda_{s+1}).$$

Next, we consider a lower bound on t^* based on λ_s :

$$(8) \quad t^* \geq \sum_{r=1}^{s-1} \mu_r + \lambda_s + \sum_{r=s+1}^k \mu_r = \sum_{r=2}^{s-1} \lambda_r + 2\lambda_s + \sum_{r=s+2}^k \lambda_r.$$

Note that we are using the convention that an empty sum is zero. Since we are working with a counterexample, we have $\frac{t}{t^*} > \frac{5m-2}{4m-1}$. This, together with the relations (8) and (6) imply

$$(9) \quad \frac{2\lambda_2 + \sum_{r=3}^k \lambda_r}{\sum_{r=2}^{s-1} \lambda_r + 2\lambda_s + \sum_{r=s+2}^k \lambda_r} > \frac{5m-2}{4m-1}.$$

If $s \in \{3, 4, \dots, k - 1\}$, then the last inequality is equivalent to

$$(10) \quad \lambda_2 > \frac{1}{3} \left(1 - \frac{1}{m} \right) \sum_{r=3}^{s-1} \lambda_r + \left(2 - \frac{1}{m} \right) \lambda_s - \frac{1}{3} \left(4 - \frac{1}{m} \right) \lambda_{s+1} + \frac{1}{3} \left(1 - \frac{1}{m} \right) \sum_{r=s+2}^k \lambda_r.$$

Finally, relations (7) and (10) imply

$$\begin{aligned} & \left(\frac{5}{6} - \frac{1}{3m} \right) \left(\sum_{r=3}^{s-1} \lambda_r + \sum_{r=s+2}^k \lambda_r \right) + \left[\frac{5}{2} - \frac{1}{m} - \frac{(5m-2)(m-2)}{2m(m-1)} \right] \lambda_s \\ & + \left(\frac{5}{3} - \left\lfloor \frac{17m-8}{3m(m-1)} \right\rfloor \right) \lambda_{s+1} < 0. \end{aligned}$$

For m and k at least four, sum of the first two terms on the left-hand-side is clearly positive. The last term is nonnegative for every $m \geq 4$. Hence, we reached a contradiction. Therefore, we may assume, $s = 2$.

Let us go back to relation (4) and use $s = 2$ and $\lambda_k \leq \lambda_4$ to obtain:

$$(11) \quad \left(\frac{m-1}{5m-2} \right) t < \frac{\lambda_4}{m} + \left(1 - \frac{2}{m} \right) (\lambda_2 - \lambda_3).$$

Substituting (6) into the above, we have

$$(12) \quad \lambda_2 > \left(\frac{6m^2 - 13m + 4}{3m^2 - 10m + 4} \right) \lambda_3 + \left(\frac{m^2 - 6m + 2}{3m^2 - 10m + 4} \right) \lambda_4.$$

Since $s = 2$, (9) becomes

$$(13) \quad \lambda_2 < \left\lfloor \frac{4m-1}{2(m-1)} \right\rfloor \lambda_3 - \frac{1}{2} \sum_{r=4}^k \lambda_r.$$

Now, using the fact that $m \geq 4$, relations (12) and (13) imply

$$\left(\frac{5m^2 + 8m - 7}{m-1} \right) \lambda_3 < -(m^2 + 2m + 8) \lambda_4 - (3m^2 - 10m + 4) \sum_{r=5}^k \lambda_r.$$

For m at least four, the coefficient of λ_3 in the left-hand-side above is positive, thus the left-hand-side is positive. However, the right-hand-side is always nonpositive. Hence, we reached a contradiction. Therefore, our original claim that Coffman–Sethi conjecture holds for all instances with m or k at least four is true. \square

Theorem 5. *The LD algorithm has a makespan ratio with a worst-case bound equal to $\frac{5m-2}{4m-1}$. Moreover, this bound is achievable for every $m \geq 2$.*

Proof. Validity of the bound follows from Theorems 3 and 4. Second statement of the theorem is established by the family of instances presented in Section 1. \square

Acknowledgment: The work was supported in part by Discovery Grants from NSERC (Natural Sciences and Engineering Research Council of Canada).

REFERENCES

- [1] Bruno, J., E. G. Coffman and R. Sethi, Algorithms for minimizing mean flow time, *INFORMATION PROCESSING '74: Processings of the IFIPS Conference*, 504–510, North-Holland Publishing Company, 1974.
- [2] Chang, S. Y. and H. C. Hwang, The worst-case analysis of the MULTIFIT algorithm for scheduling nonsimultaneous parallel machines, *Discrete Applied Mathematics*, 92(2–3) 135–147, 1999.
- [3] Coffman, E. G. M. R. Garey and D. S. Johnson, An application of bin-packing to multiprocessor scheduling. *SIAM Journal of Computing* 1, 1–17, 1978.

- [4] Coffman, E. G., M. R. Garey and D. S. Johnson, Dynamic bin packing, *SIAM Journal of Computing*, 12, 227–258, 1983.
- [5] Coffman, E. G. and R. Sethi, Algorithms minimizing mean flowtime: schedule-length properties, *Acta Informatica* 6, 1–14, 1976.
- [6] Coffman, E. G. and M. Yannakakis, Permuting elements within columns of a matrix in order to minimize maximum row sum, *Mathematics of Operations Research*, 9(3), 384–390, 1984.
- [7] Conway, R. W., W. L. Maxwell and L. W. Miller, *Theory of Scheduling*, Addison Wesley, MA, USA, 1967.
- [8] Dosa, G. Generalized multifit-type methods II, *Alkalmaz. Mat. Lapok*, 20(1), 91–111, 2000.
- [9] Dosa, G. Generalized multifit-type methods for scheduling parallel identical machines, *Pure Mathematics and Applications*, 12(3), 287–296, 2001.
- [10] Eck, B. T. and M. Pinedo, On the minimization of the makespan subject to flowtime optimality, *Operations Research*, 41(4), 797–801, 1993.
- [11] Friesen, D. K., Tighter bounds for the MULTIFIT processor scheduling algorithm, *SIAM Journal of Computing* 13, 179–181, 1984.
- [12] Garey, M. R. and D. S. Johnson, Approximation algorithms for bin packing problems—a survey, in *Analysis and Design of Algorithms in Combinatorial Optimization*, G. Ausiello and M. Lucertini, (eds.), Springer-Verlag, New York, 147–172, 1981.
- [13] Graham, R. L., Bounds for certain multiprocessing anomalies, *Bell Systems Technical Journal*, 45(9), 1563–1581, 1966.
- [14] Graham, R. L., Bounds on multiprocessing timing anomalies, *SIAM Journal of Applied Mathematics*, 17(2), 416–429, 1969.
- [15] Hochbaum, D. S. and D. B. Shmoys, Using Dual Approximation Algorithms for Scheduling Problems: Theoretical and Practical Results, *Journal of the ACM*, 34(1), 144–162, 1987.
- [16] Hsu, W.-L., Approximation algorithms for the assembly line crew scheduling problem, *Mathematics of Operations Research*, 9, 367–383, 1984.
- [17] Hwang, H.-C. and K. Lim, Exact performance of MULTIFIT for nonsimultaneous machines, *Discrete Appl. Math.*, 167, 172–187, 2014.
- [18] Johnson, D. S., Near optimal bin packing algorithms, *Ph.D. Thesis*, Mathematics Dept., MIT, Cambridge, MA, 1973.
- [19] Johnson, D. S., A. Demers, J. D. Ullman, M. R. Garey and R. L. Graham, Worst-case performance bounds for simple one-dimensional packing algorithms, *SIAM Journal of Computing*, 3, 299–325, 1974.
- [20] Ravi, P. S., *Techniques for Proving Approximation Ratios in Scheduling*, M.Math. Thesis, Dept. of Combinatorics and Optimization, Faculty of Mathematics, University of Waterloo, Canada, 2010.
- [21] Ravi, P. S., L. Tunçel and M. Huang, Worst-case performance analysis of some approximation algorithms for minimizing makespan and flowtime, *manuscript*, December 2013, revised: April 2015.
- [22] de la Vega, W. F. and G. S. Lueker, Bin packing can be solved within $1 + \epsilon$ in linear time, *Combinatorica*, 1(4), 349–355, 1981.
- [23] Yue, M. Y., On the exact upper bound for the multifit processor scheduling algorithm. *Annals of Operations Research*, 24, 1–4, 233–259, 1990.